Pattern discovery and specification techniques for Alarm Correlation

Robert D. Gardner & David A. Harle
University of Strathclyde

Communications Division
Dept. of Electrical & Electronic Eng.
University of Strathclyde
Royal College Building
204 George Street
GLASGOW G1 1XW

Tel: +44 141 548 2717
Fax: +44 141 552 4968

r.gardner, d.harle@eee.strath.ac.uk

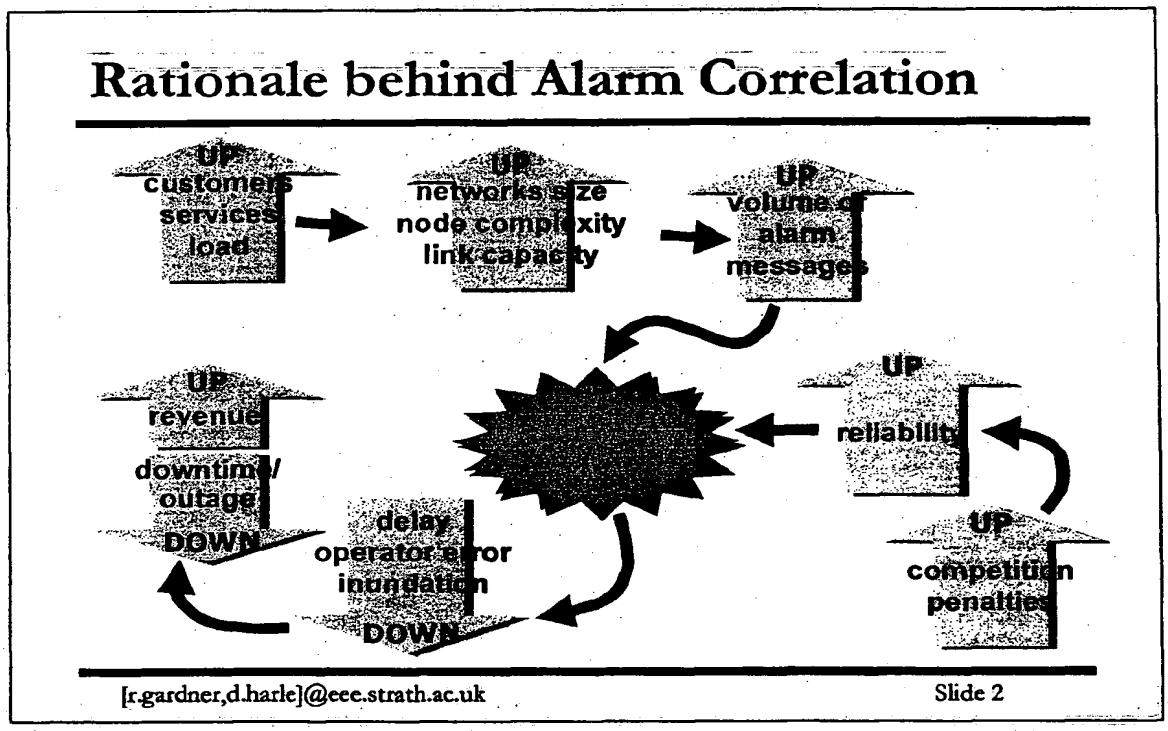
NOMS 1998

ABSTRACT

Ever increasing amounts of alarm data threaten the stability of management systems in high speed telecommunications networks. As networks continue to develop, becoming larger, using substantially higher bandwidth links and using more complex equipment, so the danger of alarm inundation is increased. Alarm correlation systems have been seen to play a vital role in dealing with the problem. However, the question of what to correlate and how to recognise and specify related alarms has either been left largely unanswered or distinct from the physical correlation process. In this presentation, we describe a unifying framework which uses a purpose-designed language to specify alarm patterns and then use the results in a real-time correlation engine. In order to test the effectiveness of the solution, the language was translated onto an existing proprietary correlation system and fed with alarm data from an SDH network test-bed. Preliminary evaluation has indicated the system to be extremely fast, potentially robust to network dynamics and, importantly, resilient to a degree of input space error. Furthermore, it is easy to use, intuitive and may be extended through the incorporation of artificial intelligence modules.

KEYWORDS

Network Management, Alarm Correlation, Data Mining, Knowledge Discovery in Databases, Telecommunications Management Network, TMN, Fault Localisation, Fault Detection, Fault Resolution, Fault Management, Fault Recognition, Pattern Matching, HP OpenView, Event Correlation Services (ECS).

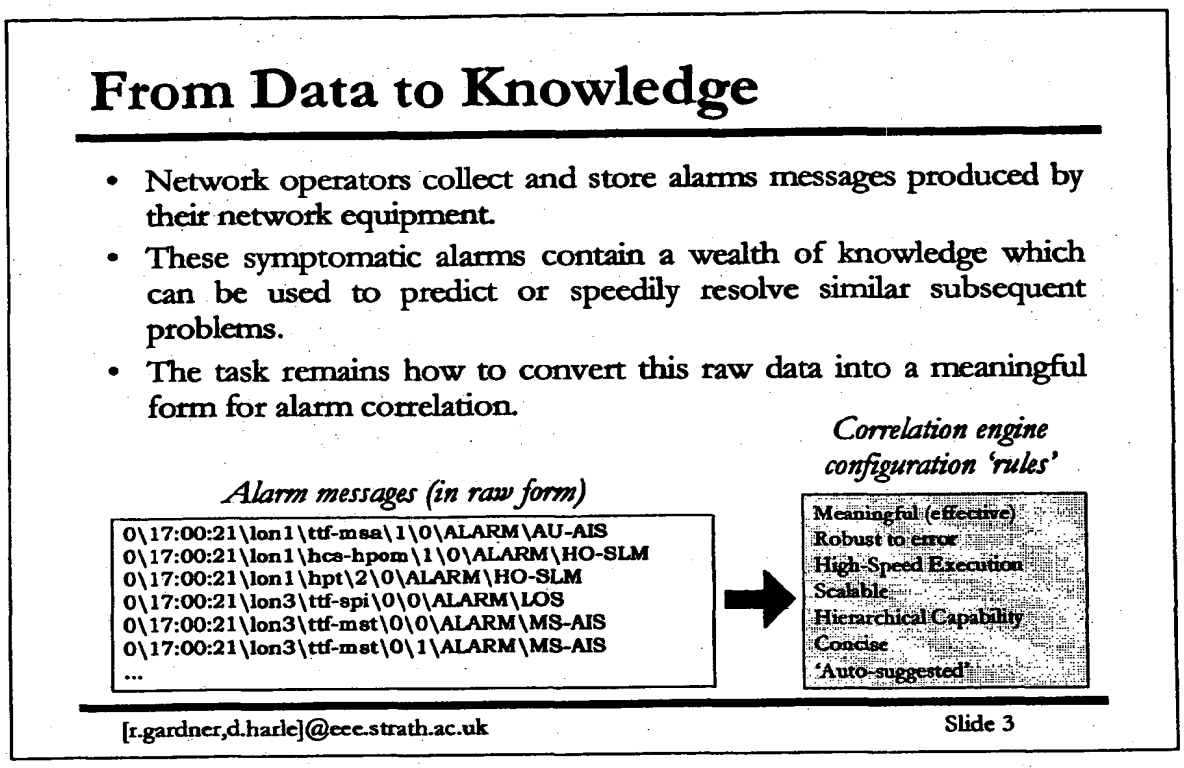


1. INTRODUCTION

The recent global expansion in the demand for telecommunications services has resulted in the considerable growth of networks in terms of size, complexity and bandwidth. Networks often consist of hundreds or even thousands of interconnected nodes from different manufacturers using various transport media and systems. As a result, when a network problem or failure occurs, it is possible that a very large volume of alarm messages is generated. Even a *typical* day on a large network can see the production of several million alarm messages. It is possible to use the alarms to help establish the nature and location of the underlying fault but at the same time, the high volume can seriously threaten an operations centre [2, 4, 5, 8]. It is therefore no surprise that alarm handling is a high priority for leading telecommunications providers.

In addition to the increasing demand, deregulation has introduced fierce competition into the industry and this has led to the need for the telecommunications operators to provide very high standards of service and reliability [14, 15]. Assisting operators in analysing the alarm can significantly reduce information and pinpointing the nature and location of faults, network downtime, which can be very costly.

In this presentation, we briefly describe the basics of a new, more unified correlation framework *(§3)* which uses a custom alarm stream language *(§4)* that can not only help discover alarm patterns but also allow the automatic configuring a real-time correlation engine. This may form the basis of a new methodology, which we briefly outline, for assisting domain experts in converting logged network alarm data into useful and meaningful knowledge. Issues such as error robustness *(§4.5)*, distributed processing and scaleable architectures *(§5)* are also considered.



2. BACKGROUND

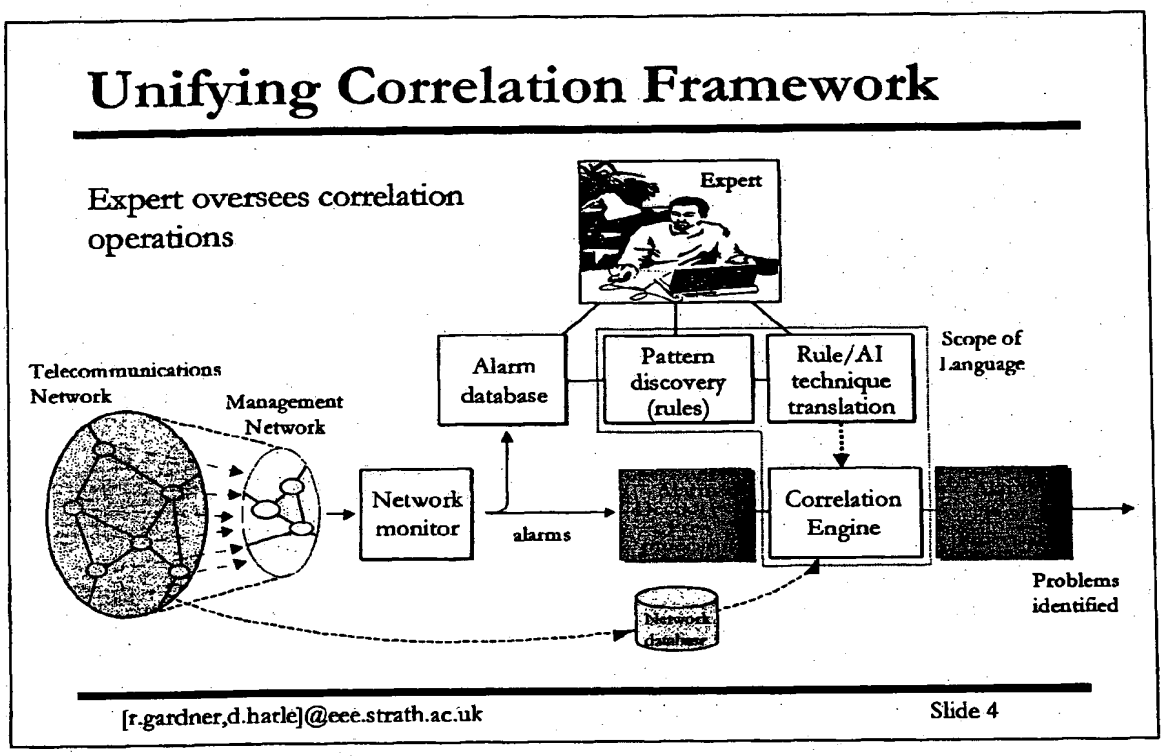
2.1 ALARMS

Alarms are short messages, generally of textual format, that are symptomatic of a change in condition (often an abnormality) in a system. Alarm messges typically contain several fields giving information about creation time, location and some condition to which the alarm pertains.

Unfortunately, alarms do not usually include explicit information regarding the exact nature of the fault. This, however, can often be inferred from the interpretation of multiple alarms generated around the time of the fault. It is important to note that although one fault may be the root cause of other faults, an alarm never causes the generation of other alarms. Other information, such as regularly generated performance messages may also prove useful in the correlation process.

2.2 ALARM CORRELATION

Alarm correlation is fast becoming the primary technique used to solve the problem of managing large volumes of event messages [4]. It involves the structured and ordered execution of a process to effect the real-time diagnosis of faults and fault localisation [10, 11]. It can reduce the amount of information presented to network operator by filtering out unnecessary or redundant alarms. The semantic content of the information presented may be simultaneously increased by the correlation and analysis processes, hence helping to establish the underlying problem or condition which produced the alarms. Successful implementation of alarm correlation systems can increase revenue since problems can be identified more quickly, resulting in a quicker restoration of service. However, too much generalisation of alarms is equally as catastrophic as too little and a balance must be achieved.



2.3 History Logs and Knowledge Therein

Network operators store old alarm messages in chronologically ordered alarm databases called logs. The logs are useful, for instance, when a fault severs communication between a network element and the correlation system. In such cases, the logged alarms may hold the only clues as to the root cause of the problem. Logs also preserve a historical record of the network performance, providing highly detailed knowledge of network faults and the alarms they produced [9]. In this respect, the logs can be invaluable in the design and configuration of correlation systems. The problem is to be able to effectively extract the pertinent information and present it in a form readable by the correlation system.

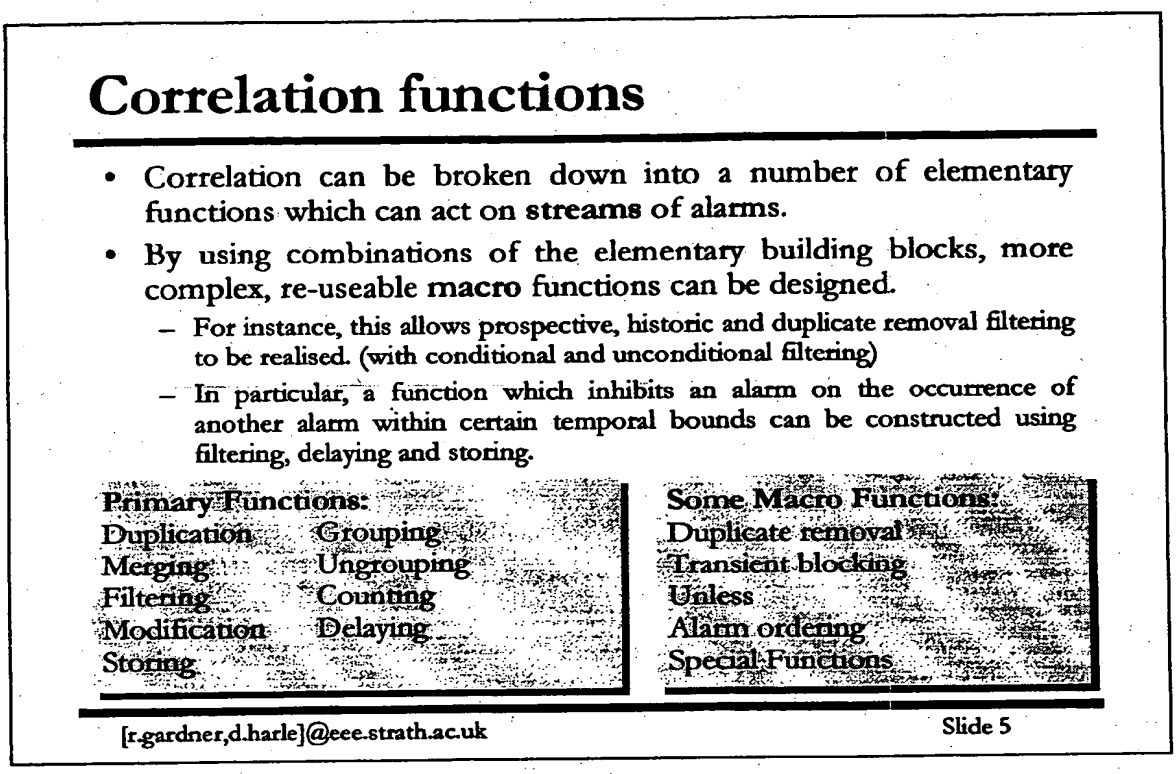
3. Unifying Correlation Framework

The structue of the proposed correlation framework is outlined in the upper half of the page. Real-time network alarms and other pertinent data (such as performance information) are passed to the correlation engine and the alarm database [12].

Key distinguishing features are pattern discovery module and integration of the robust artificial intelligence paradigms within or alongside the rule-based approach. The correlation stream language *(§4)* is used as part of a *unified* interface for the discovery of alarm features, translation into rules and the final configuration of the correlation engine. The network database contains extensive details of network and equipment structure and configuration, which may be helpful in inferring the root cause of a more subtle fault.

Before going 'on-line', the correlation engine is primed with as many alarm scenarios as is possible with the assistance of the expert, the database and the correlation stream language, which can help discover the hidden patterns. Thereafter, when the incoming alarm pattern is known, the correlation analysis provides the network operator with location and nature of the

continued on next visual...



fault or condition. If a pattern is only partly known or not known at all, the operator is presented with all available information and the 'closest' matching fault patterns - similar to the case based-reasoning paradigm. Whilst still on-line, the expert then carries out configuration maintenance instructions on the correlation engine. An option also exists to convert alarm rules, where possible, into a vector format for the artificial neural or Kohonen network features in the system [6, 7].

4. CORRELATION STREAM LANGUAGE

4.1 PRIMARY CORRELATION FUNCTIONS

Correlation functions are required to manipulate a stream of alarm messages in a structured and unhindered way. By refining the definition to involve the forming of composites, altering and discarding alarms in a stream according to their predicate information and time-stamp, we can broadly arrive at the following primary functions:

merging, duplicating, filtering, delaying, modifying
counting, storing, grouping, ungrouping.

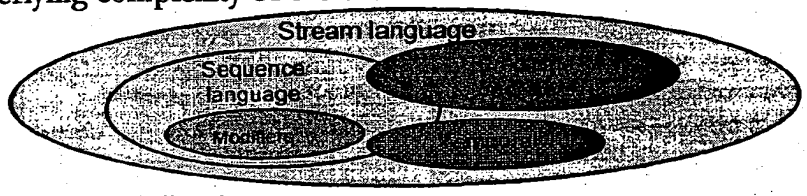
Consider relating the above functions to those through an interesting analogy with the flow of water using confluence, splitting, damming, temperature and speed of flow, for instance.

4.2 GRAMMAR

One of the primary considerations in developing a correlation language was to make it as intuitive and user-friendly as possible. One attractive feature of the language is its close grammatical connection with a modern language such as English. Instead of an obscure code, the language mimics the grammatical structure of a simple English sentence. The example in the visual #6 illustrates how a subject, a verb action, a direct object and a subordinate clause have direct equivalents in the correlation language context.

Stream Language

- The language is based on the elementary stream functions and has a grammar comparable to a spoken language and masks the underlying complexity of the basic functions.



- Compare the following:

subject	*verb*	*direct object*	*subordinate clause*
John McDonald	*| climbs*	*| the ladder*	*| if it is wooden*
stream B =	*| block*	*| stream A alarms*	*| if type is 'X'*

- Stream language vocabulary:
  block, pass, merge, substitute_alm, group, ungroup, modify_attr, add_attr, delay, count, store (also some redundancy to aid clarity: if, in, stream etc)

[r.gardner,d.harle]@eee.strath.ac.uk Slide 6

4.3 VOCABULARY

4.3.1 STREAM LANGUAGE

The main vocabulary of the correlation stream language is closely related to the primary correlation functions. Some other macro functions were considered sufficiently common and useful (or tricky to realise) to warrant their inclusion in the basic language. An important example which falls into this category is the *unless* function. This function will allow an alarm to pass unless an appropriate inhibiting alarm turns up within a certain time window. Together with the *grouping* function, a wide range of very powerful functions can be developed to groom an incoming alarm stream.

4.3.2 SEQUENCE SUB-LANGAUGE

One of the most important aspects of alarm correlation is the ability to detect patterns in an alarm stream. Whether they be consecutive alarms or otherwise, a simple and powerful sequence sub-language is required which handles almost any expressible pattern within reason. Three sequence operators are needed to perform this task and they are: **then** (= followed by), **and** (= at the same time as) and **or** (= instead of). The sequence sub-language inevitably groups single alarms into composite structures. Note that alarms within a composite can be individually accessed using a numeric subscript or by the prior ungrouping of the composite.

4.3.3 MODIFIERS

Modifiers are used in the sequence sub-language to select particular events according to their chronological relation with other alarms as opposed to any information they hold in their attribute fields. They all pertain to one or more alarms in a single stream and are: **last**, **nth**, **any** and **no** (which is signify the absence of alarms).

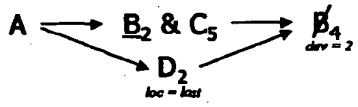
Sequence sub-language & modifiers

- To specify and identify patterns 'hidden' within streams of alarms, a powerful sequence sub-language groups related alarms.
  - **Operators** relate successive alarms using *temporal* relations:

    then and or
  - **Modifiers** can be used to pick out certain alarm messages through wildcard and definitive sequence selection:

    consecutive last nth any no
  - Optional **Field Predicates** allow alarm to be accepted or rejected on the basis of their other (non-sequence) field parameters: e.g. location
- Text or Graphical development. E.g.

A then ((consecutive B within 2 and C within 5) or D within 2, loc = last) then no B within 4, dev = 2



[r.gardner,d.harle]@eee.strath.ac.uk Slide 7

4.3.4 PREDICATES

Predicates are conditional expressions, which involve the attributes of events and return a Boolean value. They allow the attributes of events to be checked against a constant value or the comparison between the attributes of different events. Predicates are necessary in determining which events should take part in and which should be excluded from the various correlation functions of a rule sentence. Predicates may, of course, be used to merely filter a stream of passing events. The general form of a predicate expression is:

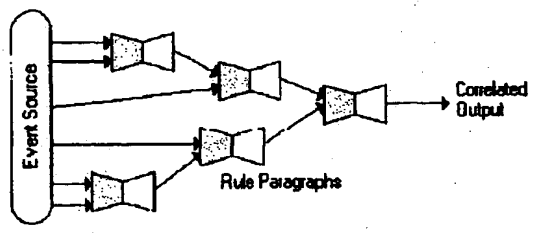
if <alarm_attribute> <operator> <alarm_attribute / constant / database value>

Some common examples of attributes are: alarm type, creation time, device location and severity of alarm. Operators include but are not limited to:

= != < > <= >= not and or exists
is_missing is_same_as is_not_same_as is_adjacent_to

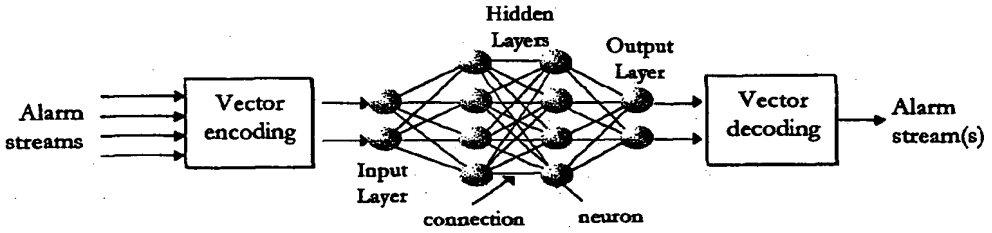
4.4 PARAGRAPHING

'Sentences' do not usually exist in isolation. A single sentence (in the correlation stream language) processes a stream of alarms to produce one or more new streams of alarms. In order to build up correlation processes, sentences must feed each other with streams of alarms. This is achieved quite simply by assigning arbitrary labels to all newly created streams. The notion of paragraphing is employed to group related sentences, which perform an identified task in the overall correlation process. Note that alarms originate from a single stream called the **SOURCE** and finally terminate at one of the **SINK**s.



Neural function integration

- *Special functions* such as artificial intelligence paradigms may be used to complement or replace regular correlation functions.
- Artificial neural networks are especially useful for identifying patterns and the generalising and classification of the data space.
- Neural functions require vector encoding/decoding of alarms using standard correlation functions such as counting and modification.



[r.gardner,d.harle]@eee.strath.ac.uk Slide 8

4.5 EXTENSION THROUGH A.I. TECHNIQUES

Artificial intelligence techniques, such as neural networks can be effectively used in the alarm correlation domain. They are adaptive systems with a parallel architecture that can learn and generalise from input data. Neural approaches are particularly useful in problems that would otherwise require complex modelling of the problem domain. Classic application areas are those involving the generalisation, categorisation and association of highly non-linear data, tasks for which they have been shown to outperform the most powerful computers.

The neural network can be trained with vector coded patterns of alarms which it will subsequently be able to 'remember' and distinguish between. Vector coding is necessary because alarms are generally of an alphanumeric form. When processing the alarms, important semantic content must not be lost or unduly obscured. Vectors can be formed on the basis of network properties such as symmetry and topological proximity [3, 6].

Neural techniques can often recognise patterns even when the input data is noisy, corrupted or has a lot of variation. This is important since it is possible for alarms to go missing or to be spuriously generated. The disadvantage is that they require intensive training before being able to associate an output pattern with a given input pattern. Although this learning process may take place, it is not always convenient in a telecommunications environment where all the alarm signatures of fault occurrences may not be known or are not amenable to vector coding.

Another form of neural network is the Self-Organising Map or Kohonen Network [7] which uses a form of competitive learning. In contrast to conventional ANNs, the learning process is unsupervised and therefore is not constrained by the existence of a correct output solution. Consequently, the primary application of an SOM is the analysis and classification of complex vectorial input space where unknown data clusters may exist. This can be of further assistance in identifying patterns of alarms either in real-time or from the alarm log.

Examples of stream processing

- Duplicate removing/Transient blocking
  - blocks similar alarms produced within a certain time

    stream B = block(stream A) if type is_duplicate within 10s
- Nearest match combiner
  - combines alarm with closest partner
  - e.g. stream $a_1,a_2,a_3,c_1,b_1,a_4,b_2,c_2$ produces $(a_3.b_1)$ $(a_4.b_2)$

    stream A = pass(source) if type = 'a'
    stream B = pass(source) if type = 'b'
    stream C = group(last in stream A then first in stream B) within 5s
- Pattern discovery
  - finds given pattern in stream of alarms
  - e.g. LOS (loss of signal) alarm followed by any other 4 alarms

    stream L = pass (source) if type = 'LOS'
    stream O = block(source) if type = 'LOS'
    stream P = group(stream L then any 4 in stream O) within 5s

[r.gardner,d.harle]@eee.strath.ac.uk Slide 9

5. CENTRALISED VS DISTRIBUTED PROCESSING

Even a simple misconfiguration or hardware fault can produce an storm of many thousands or even million of alarms. As the fault effects propagate through the network on affected paths, many of these alarms are duplicated and are hence redundant.

This highlights one need for the de-centralisation of alarm correlation entities to give regional alarm reduction capability. De-centralisation also means that any parts of the network which become temporarily unreachable via management channels may still survive by dint of their localised correlation and fault protection systems.

It makes sense, however, for the distributed processes to be administered and maintained by a team of centralised network experts can share facilities, knowledge and their experiences.

6. TESTING THE LANGUAGE USING SDH ALARMS

In order to help verify the effectiveness of the language, a transmission system simulator based on the Synchronous Digital Hierarchy was used. The simulation consisted of 11 add-drop multiplexers arranged as a set of interconnected rings with STM-4 (620 Mbit/s) link capacities.

Various faults were introduced on the simulated network and the alarms produced collected to form an alarm log. The correlation stream language was used interactively to capture and specify distinct fault signatures, which were then be automatically compiled onto a proprietary correlation engine (Hewlett-Packard's Event Correlation Services (ECS) forming part of their OpenView telecommunications management platform) [1, 13]. The correlation engine was then able to handle real-time alarm data.

The procedure demonstrated the ease and speed at which fault signatures could be discovered and integrated into a correlation system. By providing a common interface to all the accessible modules of the system, the correlation language simplified and helped standardise a methodology for the expert-machine communication process.

7. SUMMARY & CONCLUSIONS

Effective Network Management leads to greater efficiency, higher profits, better reputation, more flexibility, higher reliability and ease of maintenance. In this presentation, we have briefly described a unified framework using a powerful correlation language which can assist network operators achieve such goals.

The components of the correlation language allow correlation procedures to be designed and implemented regardless of the underlying technology. Furthermore, the complementary use of artificial intelligence building blocks improves error resilience and adds generalisation functionality to the more declarative language.

Testing showed the expert system to be easy to use, fast, flexible and amenable to hierarchical management strategies by dint of scalability. The system in no way attempts to replace the job of the network expert. Rather it focuses on increasing efficiency with the help of the expert providing a structured and traceable approach to the problem.

ACKNOWLEDGEMENTS

The authors gratefully acknowledge the help of the Toccata project (especially Keith Harrison, Michele Campriani and John Manley) of Hewlett-Packard Laboratories, Bristol, England in providing technical support and funding for this work.

REFERENCES

1. Harrison, K.A., INCL, Hewlett-Packard Bristol Laboratories, Event Correlation in Telecommunications Network Management, 20-09-1994.
2. R.Gardner & D.A.Harle, Methods and Systems for Alarm Correlation, proceedings of Globecom'96, London, November 18-22, 1996, pp.136-140
3. R.Gardner & D.A.Harle, Using Network Configuration Information in Alarm Correlation, Research Report, University of Strathclyde, Communications Division.
4. Jakobson, G., Weissman M., Alarm Correlation, IEEE Network, Vol. 7, No. 6.
5. Jakobson, G., Weissman M., Real-time telecommunication network management: extending event correlation with temporal constraints, Integrated Network Management - proceedings of the fourth international symposium on integrated network management 1995, pp. 290-301, Chapman & Hall, ISBN 0-412-71570-8, 1995.
6. R.Gardner & D.A.Harle, Network Fault Detection: A Simplified Approach to Alarm Correlation - proceedings of IEEE International Switching Symposium September 1997, Toronto, Canada.
7. R.Gardner & D.A.Harle, Alarm Correlation and Network Fault Resolution using the Kohonen Self-Organising Map - proceedings of IEEE Globecom'97, November 4-8, 1997, Phoenix, Arizona, USA.
8. R.Gardner & D.A.Harle, Expert Data Mining for Alarm Correlation in High-Speed Networks - proceedings of IITT EXPERSYS'97, October 1997, Sunderland, UK.
9. Salah Aidarous, Thomas Plevyak (editors), Telecommunications Network Management into the 21st Century, IEEE Press, 1994, ISBN 0-85296-814-0.
10. S.Kliger, S.Yemini, A Coding Approach to Event Correlation, Integrated Network Management - proceedings of the fourth international symposium on integrated network management 1995, pp. 266-277, Chapman & Hall, ISBN 0-412-71570-8, 1995.
11. A.T.Bouloutas, S.Calo, A.Finkel, Alarm Correlation and Fault Identification in Communication Networks, IEEE Transactions on Communications, Vol.42, No.2/3/4, Feb/Mar/Apr 1994.
12. Masoud Mansouri-Samani, Morris Sloman, Monitoring Distributed Systems, Imperial College Of Science Technology and Medicine, Department of Computing, London.
13. Hewlett-Packard Company, Event Correlation Services Designer's Reference, 24 November 1995.
14. Stevenson, Douglas W., Network Management: What it is and what it isn't, WWW document, Network Management Archive Server, University at Buffalo, NY, USA
15. Roch H. Glitho, Stephen Hayes, Telecommunications Management Network: Vision vs Reality, IEEE Communications Magazine, March 1995.